FAX TRANSMITTAL COVER SHEET

CONLEY ROSE, P.C.
5601 Granite Parkway, Suite 750
Plano, Texas 75024-6616
Fax Number: (972) 731-2289
Telephone Number: (972) 731-2288

PLEASE DELIVER THE FOLLOWING PAGES IMMEDIATELY TO:

NAME: Examiner Isaac Tuku Tecklu, Group Art Unit 2192
U.S. Patent and Trademark Office

FAX: 571 273 9957

FROM: John Leonard (sent by Jean Brown)

DATE: May 28, 2009

RE: U.S. Patent Application No. 10/820,539
Applicant Initiated Interview Request Form (1 page)
Interview Agenda – Not for Entry Attachment (26 pages)

REMARKS:

Total Number of Pages (Including This One): 28
OUR CLIENT/MATTER NO.: 4000-18700

YOUR REFERENCE NO.: Patent Application No. 10/815,518

IF YOU DO NOT RECEIVE ALL THE PAGES,
PLEASE CALL 972-731-2288 AS SOON AS POSSIBLE

PTOL-413A (12-06)
Approved for use through 01/31/2009. OMB 0651-0031
U.S. Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

Applicant Initiated Interview Request Form

Application No.: 10/820,539 First Named Applicant: Robin D. Katzer

Examiner: Isaac Tuku Tecklu Art Unit: 2192 Status of Application: Non-Final Rej.

Tentative Participants:

(1) Isaac Tuku Tecklu (2) Michael Piper

(3) Brain Genco (4) _____

Proposed Date of Interview: Tues June 2, 2009 **Proposed Time:** 4:00 E.T. **AM/PM**

Type of Interview Requested:

(1) ☐ Telephonic (2) ☑ Personal (3) ☐ Video Conference

Exhibit To Be Shown or Demonstrated: ☐ YES ☑ NO

If yes, provide brief description: _____

Issues To Be Discussed

Issues (Rej., Obj., etc)	Claims/ Fig. #s	Prior Art	Discussed	Agreed	Not Agreed
(1) Rej.	1-14, 16-23	Furrer	☐	☐	☐
(2) _____	_____	_____	☐	☐	☐
(3) _____	_____	_____	☐	☐	☐
(4) _____	_____	_____	☐	☐	☐

☐ Continuation Sheet Attached

Brief Description of Argument to be Presented:

See attached.

An interview was conducted on the above-identified application on _____ .

NOTE: This form should be completed by applicant and submitted to the examiner in advance of the interview (see MPEP § 713.01).

This application will not be delayed from issue because of applicant's failure to submit a written record of this interview. Therefore, applicant is advised to file a statement of the substance of this interview (37 CFR 1.133(b)) as soon as possible.

Applicant/Applicant's Representative Signature

John Leonard

Typed/Printed Name of Applicant or Representative

58,144

Registration Number, if applicable

Examiner/SPE Signature

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	Robin D. Katzer, et al.	§	Group Art Unit: 2192
Serial No.:	10/820,539	§	Examiner: Tecklu, Isaac Tuku
Filed:	April 8, 2004	§	Confirmation No.: 8371
For:	Integration of COTS Software Data Stores Into Integrated Data Access Layer	§	

Interview Request Attachment

Applicants acknowledge receipt of the Office Action dated March 10, 2009, and respectfully request the following proposed amendments and arguments be considered for discussion in a telephone interview on a date and time to be determined. The amendments and arguments included herein are not entered as a response to the Office Action dated March 10, 2009. The changes made are shown by underlining the added text and striking through the deleted text.

Proposed Claim Amendments are reflected in the listing of claims which begins on page 2 of this paper.

Remarks/Arguments begin on page 12 of this paper.

***Listing of the Proposed Claim Amendments*:**

1. (Currently Amended) A system for decoupling commercial-off-the-shelf software applications from data stores, the system comprising:

a plurality of commercial-off-the-shelf software applications each compatible with one of a plurality of first data stores, each of the plurality of commercial-off-the-shelf software applications submits a data request compatible with one of the plurality of first data stores;

a plurality of second data stores;

a plurality of drivers, wherein each of the plurality of first data stores and the plurality of second data stores has a corresponding one of the plurality of drivers configured to receive the data request and pass the data request to the corresponding data store;

at least one processor;

one of a plurality of listeners, recorded on a computer readable medium, when executed by at least one processor, simulates a corresponding one of the plurality of drivers corresponding with one of the plurality of first data stores and receives the data request from a corresponding one of the plurality of commercial-off-the-shelf software applications that is compatible with the one of the plurality of first data stores simulated by the one of the plurality of listeners, wherein each of the plurality of commercial-off-the-shelf software applications has a corresponding one of the plurality of listeners;

a translator, recorded on a computer readable medium, in communication with the one of the plurality of listeners and the plurality of second data stores, the translator, when executed by at least one processor, receives the data request from the one of the plurality of listeners, translates the data request, and submits the translated data request for one of the plurality of drivers corresponding with one of the plurality of second data stores for storage by the one of the plurality of second data stores.

2. (Previously Presented) The system of Claim 1, wherein the translator translates the data request into a generic format, and further comprising a data access layer, recorded on a computer readable medium, in communication with the translator and, when executed by at least one processor, determines to direct the data request from one of the commercial-off-the-shelf software applications to the one of the plurality of second data stores, and translates the translated data request from the generic format into a storage format of the one of the plurality of second data stores.

3. (Previously Presented) The system of Claim 2, wherein the data access layer maintains an enterprise data model including a data map of where to direct the data request of each of the commercial-off-the-shelf software applications.

4. (Previously Presented) The system of Claim 3, wherein the data access layer receives the translated data request from the translator and directs the translated data request to one of the plurality of second data stores.

5. (Previously Presented) The system of Claim 2, wherein a first commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications submits a first data request in a first relational database format and wherein the data access layer translates the first data request to a second relational database format.

6. (Previously Presented) The system of Claim 5, wherein a second commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications submits a second data request in an older version of the first relational database format and wherein the data access layer translates the second data request to a newer version of the first relational database format.

7. (Previously Presented) The system of Claim 2, wherein a first commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications submits a first data request in an older version of a first relational database format and wherein the data access layer translates the first data request to a newer version of the first relational database format.

8. (Previously Presented) The system of Claim 1, wherein at least one of the second data stores corresponds with one of the plurality of first data stores.

9. (Original) The system of Claim 8, wherein the at least one of the second data stores is further defined as a newer version data store of one of the plurality of first data stores.

10. (Previously Presented) The system of Claim 9, wherein at least one of the second data stores is further defined as a newer version of a relational database of a first vendor and wherein one of the plurality of first data stores is further defined as an older version of the relational database of the first vendor.

11. (Previously Presented) The system of Claim 9, wherein at least one of the second data stores is further defined as a newer version of a relational database of a second vendor and wherein one of the plurality of first data stores is further defined as an older version of the relational database of the second vendor.

12. (Previously Presented) The system of Claim 1, wherein the plurality of commercial-off-the-shelf software applications are each operable with only one of a plurality of data stores, each of the plurality of commercial-off-the-shelf software applications submitting data requests compatible with only one of the plurality of data stores.

13. (Previously Presented) A system for maintaining compatibility of commercial-off-the-shelf software applications with data stores, the system comprising:

a commercial-off-the-shelf software application operable with only a first data store, the commercial-off-the-shelf software application submits a data request compatible with only the first data store;

a first driver configured to receive the data request and pass the data request to the first data store;

at least one processor;

a listener, recorded on a computer readable medium, when executed by at least one processor, simulates the first driver and receives the data request from the commercial-off-the-shelf software application submitted for the first driver;

a translator, recorded on the computer readable medium, in communication with the listener, when executed by at least one processor, receives the data request from the listener and translates the data request into a generic format to produce a first translated data request;

a data access layer, recorded on the computer readable medium, in communication with the translator and, when executed by at least one processor, determines, based on an enterprise data model, to direct the data request of the commercial-off-the-shelf software applications to a second data store and translates the first translated data request from the generic format into a storage format of the second data store to produce a second translated data request;

a wrapper, recorded on the computer readable medium, when executed by at least one processor, receives the second translated data request from the data access layer and wraps the second translated data request based on the storage format of the second data store

a second driver configured to receive the wrapped second translated data request and pass the wrapped second translated data request to the second data store; and

the second data store receives the wrapped second translated data request from the second driver and performs an action specified in the data request.

14. (Previously Presented) The system of Claim 13, wherein the second data store is one of a newer version data store of the first data store and a different vendor database than the first data store.

15. (Cancelled)

16. (Currently Amended) A system for integration of commercial-off-the-shelf software applications and databases, the system comprising:

a plurality of commercial-off-the-shelf software applications each compatible ~~operable~~ with one of a plurality of first data stores, each of the commercial-off-the-shelf software applications submits a data request compatible with one of the plurality of first data stores;

a first driver configured to receive the data request and pass the data request to the first data store;

at least one processor;

one of a plurality of listeners, recorded on a computer readable medium, when executed by at least one processor, simulates the first driver and receives the data request from a corresponding one of the commercial-off-the-shelf software applications submitted for the first driver, wherein each of the plurality of commercial-off-the-shelf applications has a corresponding one of the plurality of listeners;

a translator, recorded on a computer readable medium, in communication with the one of the plurality of listeners, when executed by at least one processor, receives the data request from the one of the plurality of listeners and translates the data request;

a second driver configured to receive the translated data request and pass the translated data request to a second data store; wherein the second data store receives the translated data request from the second driver and performs an action specified in the data request; and

a service broker, recorded on the computer readable medium, when executed by at least one processor, maintains a record of data requests from the commercial-off-the-shelf software application and stored in the second data store, the service broker further configured to roll-back failed data requests.

17. (Previously Presented) The system of Claim 16, wherein the translator translates the data request into a generic format, and further comprising a data access layer, recorded on a computer readable medium, in communication with the translator and, when executed by at least one processor, determines, based on an enterprise data model, to direct the data request from one of the commercial-off-the-shelf software applications to the second data store, and translates the translated data request from the generic format into a storage format of the second data store.

18.(Previously Presented) The system of Claim 16, wherein the commercial-off-the-shelf software application is operable with only the first data store, and wherein the commercial-off-the-shelf software application submits the data request compatible with only the first data store.

19. (Previously Presented) The system of Claim 16, wherein the service broker further comprises:

a transaction data store configured that maintains a record of the data request by the commercial-off-the-shelf software application;

an exception handler that identifies a failed transaction and communicates with the transaction data store to restore the second data store to a state prior to the failed transaction.

20. (Previously Presented) The system of Claim 19, further comprising a data warehouse, recorded on the computer readable medium, and wherein the data warehouse, when executed by at least one processor, is asynchronously updated with the data request from the commercial-off-the-shelf software application.

21. (Original) The system of Claim 19, wherein a compensating transaction is used to restore the failed transaction.

22. (Original) The system of Claim 21, wherein an XA transaction is used in combination with the compensating transaction to restore the failed transaction.

23. (Previously Presented) The system of Claim 19, further comprising:

a data warehouse, recorded on the computer readable medium, when executed by at least one processor, maintains data;

a query processor, recorded on the computer readable medium, when executed by at least one processor, manages transaction processing of data requests from the commercial-off-the-shelf software application; and

a metadata repository, recorded on the computer readable medium, when executed by at least one processor, maintains a logical data model related to the data, wherein the metadata repository instructs the query processor regarding handling of the data requests from the commercial-off-the-shelf software application and between the second data store and the data warehouse.

Interview Agenda

This application has been carefully considered in connection with the Examiner's Office Action dated March 10, 2009. Reconsideration and allowance are respectfully requested in view of the following.

Response to Rejections

Furrer does not anticipate that a translator translates data requests into a generic format for a data access layer that directs the data requests to an appropriate data store for fulfilling the data requests and further translates the data request from the generic format into the format of the appropriate data store. Furrer also does not anticipate that each commercial-off-the-shelf software application has a corresponding listener and a listener simulates a driver corresponding with a data store and receives a data request from a commercial-off-the-shelf software application that is compatible with the data store simulated by the listener. Translating the data requests into a generic format that the data access layer can translate into the format of the appropriate data store for fulfilling the data requests enables data stores different from the data stores for which the applications were originally certified to fulfill the data requests. Simulating drivers by listeners decouples the commercial-off-the-shelf applications from the data stores that fulfill its data requests by enabling the commercial-off-the-shelf applications to submit data requests in its normal manner regardless of the characteristics of the data stores that fulfill its data requests. Therefore, the data stores that fulfill the data requests may be changed or upgraded as needed without the need for any changes in the commercial-off-the-shelf application.

Furrer teaches that a "transaction converter, interface, and results converter cooperate such that relational data requests sent by the web-application access the non-relational data through the RRM [recoverable resource manager] and return relational results from non-relational results provided by the RRM. (Abstract) Although Furrer may disclose a translator that translates a data request, Furrer does not anticipate that a translator translates data requests into a generic format for a data access layer that directs the data requests to an appropriate data store for fulfilling the data requests and further translates the data request from the generic format into the format of the appropriate data store. Furthermore, while Furrer may disclose connectors that function much like drivers, Furrer does not anticipate that each commercial-off-the-shelf software application has a corresponding listener and a listener simulates a driver corresponding with a data store and receives a data request from a commercial-off-the-shelf software application that is compatible with the data store simulated by the listener.

These and other distinctions between the pending specification and the applied art will be discussed in greater detail in the analysis of the pending claims that follows.

Response to Rejections under Section 102

Claim 13:

Claim 13 was rejected under 35 USC § 102(e) as being anticipated by Furrer et al., U.S. Pub. No. 2005/0192962 A1 (hereinafter "Furrer").

I. Furrer does not anticipate a translator that translates a data request into a generic format or a data access layer that directs the data request from a commercial-off-the-shelf software application to the appropriate data store and translates the translated data request from the generic format into the storage format of the appropriate data store.

Claim 13 recites:

> a translator . . . [that] receives the data request from the listener and translates the data request into a generic format to produce a first translated data request; a data access layer . . . [that] determines, based on an enterprise data model, to direct the data request of the commercial-off-the-shelf software applications to a second data store and translates the first translated data request from the generic format into a storage format of the second data store to produce a second translated data request

In reference to the rejection of independent Claim 13, page 9 of the Office Action cites paragraph 83 of Furrer to allege that Furrer anticipates these Claim 13 limitations.

Paragraphs 81-83 of Furrer disclose:

> Preferably, the connectors 606 are configured to bridge between legacy data management systems such as EIS 604 and modern technologies such as web-applications 610 and/or web components 610 . . . The connectors 606 comprise an technology insulation layer between web-application 610 and Application/web-server 602 technology on one side and legacy applications, operating systems, and/or system calls on the other. Consequently, the connectors 606 function much like software drivers insulate an application from particular hardware commands. Different types of connectors 606 may be implemented. For JDBC connectors 606, four different well known types exist. Type-1 connectors comprise a JDBC-ODBC bridge. Type-2 connectors comprises a native API combined with a driver written partially in the JAVA programming language. A type-2 connector converts JDBC calls into database or operating system specific data requests. In certain embodiments, the VSAM connector 606a comprises a type-2 connector written in JAVA and a language compatible with the host operating system 612 such as the z/OS from IBM. A type-3 connector comprises a driver written completely in JAVA that passed JDBC requests through a network to a middle-tier server which then translates the JDBC request into a data store specific

> data request. A type-4 connector is written completely in JAVA and converts JDBC calls into a specific database management system protocol (DBMS) for direct communication with the DBMS server.

Based on the disclosure cited above, Furrer teaches that the four types of connectors described in cited paragraph 83 are configured to bridge between legacy data management systems and web-applications. Each of the four types of connectors described in cited paragraph 83 receive Java Database Connectivity calls from an application and convert the Java Database Connectivity calls into a request for a specific database. Paragraph 51 of Furrer discloses that the cited "connector 300 includes a web interface 310, a transaction converter 312, an interface 314, and a results converter 316." Paragraph 19 of Furrer discloses the function of the connector's transaction converter and results converter by teaching that the "transaction converter converts relational transaction requests from the web-application into one or more non-relational transaction requests . . . The results converter converts the non-relational results into relational results that can be sent back to the web-application" Therefore, Furrer's connectors receive requests from relational data applications that are intended for a relational database and incompatible with a specific non-relational database, and convert the requests into specific non-relational requests that are compatible with the specific non-relational database. Cited paragraph 83 offers examples of specific non-relational databases for which requests are converted, such as "ODBC" (Open Database Connectivity), "database or operating system specific data requests . . . such as the z/OS from IBM," "a data store specific data request," and "a specific database management system protocol (DBMS) for direct communication with the DBMS server."

Paragraph 54 of Furrer discloses that after the connector converts a request into a specific non-relational request that is compatible with a specific non-relational database,

> The interface 314 sends the non-relational transaction requests provided by the transaction converter 312 to the RRM 210. As mentioned above, typically, each connector 300 corresponds with a single RRM 210 . . . The RRM 210 executes the non-relational transaction requests as though the requests came from any other legacy application requesting transactional recovery and access

Paragraph 18 of Furrer discloses that the "recoverable resource manager (RRM) provides transactional recovery and transactional access for a plurality of transactions concurrently accessing the data. Preferably, the RRM is configured for a specific type of data such as Virtual Storage Access Method (VSAM) data." Paragraph 45 of Furrer specifies "a separate recoverable resource manager (RRM) 210 for each type of legacy data. A recoverable resource manager 210 comprises a software module designed specifically to manage the specific type of legacy data." Therefore, after the connector converts a request into a specific non-relational request that is compatible with a specific non-relational database, the connector's interface sends the specific non-relational transaction request to the specific recoverable resource manager that is configured for the specific non-relational database.

In contrast, the pending disclosure teaches a translator that translates a data request into a generic format, and a data access layer that translates the data request from the generic format into the storage format used for the data store to which the data access layer is directing the translated data request:

> The translators 42, 44, and 46 translate data requests between the formats of the listeners 32, 34, and 36 and a single or generic format. The translators 42, 44, and 46 transfer the generic format data requests to and from a single data access layer 60 . . . The data access layer 60 receives data requests from the translators 42, 44, and 46 in the single or generic format into which all, or perhaps only some groups, of the translators 42, 44, and 46 have the capability of converting data requests. This single or generic format may be, for example, a language or data format or standard compatible with the all, or groups, of the translators 42, 44, and 46. Upon receiving a data request, the data access layer 60 determines which data store 92, 94, or 96 the data request should be sent to. The data access layer 60 then translates the data request from the generic format into the format of the appropriate data store 92, 94, or 96. The data access layer 60 then sends the data request to the wrapper 72, 74, or 76 for the appropriate data store 92, 94, or 96. The data access layer 60 can translate data requests from the generic language provided by the translators 42, 44, and 46 into the native language of each data store 92, 94, and 96 . . . Therefore, the data access layer 60 can actually receive a data request from any COTS application 12, 14, or 16 and send the request to any data store 92, 94, or 96. (Paragraphs 20, 22, and 25)

In contrast to Furrer, which discloses connectors that translate relational data requests into specific non-relational data requests for specific data stores, the pending disclosure teaches a listener that translates a data request into a non-specific generic format. In further contrast to Furrer, which discloses an interface that sends the translated data request only to the specific non-relational data store for which the relational data request has been translated, the pending disclosure teaches a data access layer that determines to which of many data stores to direct the data request after the data request is translated into a generic format. The data access layer determines the specific format to translate the generic data request after the data access layer identifies to which specific data store the data request is to be directed.

Accordingly, Furrer does not anticipate a translator that translates a data request into a generic format or a data access layer that directs the data request from a

commercial-off-the-shelf software application to the appropriate data store and translates the translated data request from the generic format into the storage format of the appropriate data store.

For at least the reasons established above in section I, Applicants respectfully submit that independent Claim 13 is not anticipated by Furrer and respectfully requests allowance of this claim.

Claims Depending from Claim 13:

Claim 14 was rejected under 35 USC § 102(e) as being anticipated by Furrer.

Dependent claim 14 depends directly from independent claim 13 and incorporates all of the limitations thereof. Accordingly, for at least the reasons established in section I above, Applicants respectfully submit that claim 14 is not anticipated by Furrer and respectfully request allowance of these claims.

Claim 1:

Claim 1 was rejected under 35 USC § 102(e) as being anticipated by Furrer.

I. Furrer does not anticipate that each commercial-off-the-shelf software application has a corresponding listener and a listener simulates a driver corresponding with a data store and receives a data request from a commercial-off-the-shelf software application that is compatible with the data store simulated by the listener.

Amended Claim 1 recites:

> one of a plurality of listeners . . . simulates a corresponding one of the plurality of drivers corresponding with one of the plurality of first data stores and receives the data request from a corresponding one of the plurality of commercial-off-the-shelf software applications that is compatible with the one of the plurality of first data stores simulated by the one of the plurality of listeners, wherein each of the plurality of commercial-off-the-shelf software applications has a corresponding one of the plurality of listeners

Applicants respectfully submit that no new matter has been introduced by the amendments to claim 1. Support may be found throughout the specification as originally filed, including Figures 1and 2, and paragraphs 18 and 19.

In reference to a rejection of independent Claim 1, the Office Action alleges on page 3 and 4 that:

> Furrer discloses . . . a listener . . . simulates one of the plurality of drivers corresponding with one of the plurality of first data stores and receive the data request from one of the plurality of commercial-off-the-shelf software applications that is compatible with the one of the plurality of first data stores simulated by the listener (paragraph [0083] ". . . JDBC-ODBC bridge . . . API combined with a driver . . . driver written completely in JAVA that passed JDBC request . . . middle-tier server . . .")

As discussed above in section I, Furrer teaches that the four types of connectors described in cited paragraph 83 are configured to bridge between legacy data management systems and web-applications. Each of the four types of connectors described in cited paragraph 83 receive Java Database Connectivity calls from an application and convert the Java Database Connectivity calls into a request for a specific database. Paragraph 51 of Furrer discloses that the cited "connector 300 includes a web interface 310, a transaction converter 312, an interface 314, and a results converter 316." Paragraph 52 of Furrer discloses the function of the connector's web interface by teaching that

> the web interface 310 receives a relational transaction request from a web-application 212. The web interface 310 comprises a published interface for receiving relational transaction requests. Preferably, the published interface is an industry-accepted application programming interface (API) such as Java Database Connectivity (JDBC), ODBC, or the like. Consequently, the relational transaction request is formatted according to the SQL protocol. Of course, the API of the web interface 310 may be modified to accommodate new data request protocols without changing other components of the VSAM connector 300.

Therefore, Furrer's connectors include a web interface that may be a Java Database Connectivity application programming interface that receives requests formatted according to the SQL protocol. Although the web interface may be modified to accommodate different data request protocols, Furrer does not anticipate multiple connectors that each receives requests formatted according to different data request protocols. The abstract of Furrer specifies that an apparatus, system and method are provided "such that relational data requests sent by the web-application access the non-relational data through the RRM [recoverable resource manager] and return relational results from non-relational results provided by the RRM." Therefore, Furrer teaches that the cited connectors receive relational data requests.

Paragraph 62 of Furrer specifies that although the cited connectors function as an interface between legacy databases and the web applications that send relational data requests, the legacy applications bypass the cited converters to access the legacy databases:

> Recoverable relational transactions from web-applications 402 are made possible by the connector 300 in conjunction with the RRM 406, RRS 408, and CAF 410. Of course, the batch and legacy applications 412, 414 send non-relational transaction requests directly to the RRS [resource recovery service] 408.

Furrer discloses that a legacy application may send a non-relational data request to its corresponding legacy data store, and that the connectors that enable relational data requests to access non-relational legacy data stores do not receive any non-relational data requests from legacy applications. Because paragraph 5 of Furrer discloses that "the legacy data storage subsystem is simply a file system of an operating system that manages a number of different files storing data in proprietary formats," one legacy application may not be able to access the data stored in a proprietary format by another legacy data store. Furrer does not anticipate any connectors that correspond to the legacy applications to enable the different legacy applications to access each others' corresponding legacy data stores. Cited paragraph 83 of Furrer teaches that the cited connectors correspond to the various data stores, such as "ODBC" (Open Database Connectivity), "database or operating system specific data requests . . . such as the z/OS from IBM," "a data store specific data request," and "a specific database management system protocol (DBMS) for direct communication with the DBMS server." However, the cited connectors do not correspond to the applications that may access the data stores, particularly to the legacy applications that correspond to the legacy data stores.

In contrast, the pending disclosure teaches that each commercial-off-the-shelf software application has a corresponding listener and a listener simulates a driver corresponding with a data store and receives a data request from a commercial-off-the-shelf software application that is compatible with the data store simulated by the listener. Paragraphs 19 and 27 of the pending disclosure clearly differentiate between the listeners that correspond to each of the applications the pending disclosure and the

connectors of Furrer. Paragraphs 19 and 27 of the pending disclosure are reproduced below:

> The listeners 32, 34, and 36 simulate the functions of the listeners that are typically built in to commercially available data stores. More specifically, each listener 32, 34, or 36 simulates the listener of the data store for which its corresponding COTS application 12, 14, or 16 was certified. Thus, a COTS application 12, 14, or 16 can submit a data request in its normal manner and, from its perspective, the data request is received by the same listener that normally receives its data requests.
>
> The SQL statement is received by the MS SQL Server listener 32. The MS SQL Server listener 32 simulates the functions of the listener on the data store for which the COTS application 12 has been certified. This gives the COTS application 12 the impression that it is submitting the SQL statement to the data store for which it has been certified.

The pending disclosure teaches that each listener simulates the listener of the data store for which its corresponding commercial-off-the-shelf application was certified, and that each commercial-off-the-shelf application has a corresponding listener. Although Furrer may disclose that each non-relational legacy data store has a corresponding legacy application, each legacy application does not have a corresponding connector because the legacy application bypasses the connectors to access the legacy data stores. In contrast to Furrer' connector, which receives a relational data request for a non-relational data store that is incompatible with the relational application that sent the relational data request, the listener taught by the pending disclosure simulates a driver that receives a data request for a database that is certified as compatible with the commercial-off-the-shelf application that sent the data request.

Accordingly, Furrer does not anticipate that each commercial-off-the-shelf software application has a corresponding listener and a listener simulates a driver

corresponding with a data store and receives a data request from a commercial-off-the-shelf software application that is compatible with the data store simulated by the listener.

For at least the reasons established above in section I, Applicants respectfully submit that independent Claim 1 is not anticipated by Furrer and respectfully requests allowance of this claim.

Claims Depending from Claim 1:

Claims 2-12 were rejected under 35 USC § 102(e) as being anticipated by Furrer.

Dependent claims 2-12 depend directly or indirectly from independent claim 1 and incorporate all of the limitations thereof. Accordingly, for at least the reasons established in section II above, Applicants respectfully submit that claims 2-12 are not anticipated by Furrer and respectfully requests allowance of these claims.

Dependent Claim 2 includes limitations substantially similar to the limitations discussed in section I above. For example, Claim 2 recites:

> wherein the translator translates the data request into a generic format, and further comprising a data access layer . . . [that] determines, based on an enterprise data model, to direct the data request from one of the commercial-off-the-shelf software applications to the second data store, and translates the translated data request from the generic format into a storage format of the second data store.

Accordingly, the arguments of section I are hereby repeated for dependent Claim 2.

In addition to the reasons established in section II above, for at least the reasons established in section I above, Applicants respectfully submit that all of the limitations of

dependent Claim 2 are not anticipated by Furrer and respectfully request allowance of this claim.

Claim 16:

Claim 16 was rejected under 35 USC § 102(e) as being anticipated by Furrer.

Claim 16 includes limitations substantially similar to the limitations discussed in section II above. For example, Claim 16 recites, "one of a plurality of listeners . . . simulates the first driver and receives the data request from a corresponding one of the commercial-off-the-shelf software applications submitted for the first driver, wherein each of the plurality of commercial-off-the-shelf applications has a corresponding one of the plurality of listeners." Accordingly, the arguments of section II are hereby repeated for claim 16.

For at least the reasons established above in section II, Applicants respectfully submit that independent claim 16 is not anticipated by Furrer and respectfully request allowance of this claim.

Claims Depending from Claim 16:

Claims 17-21 and 23 were rejected under 35 USC § 102(e) as being anticipated by Furrer.

Dependent claims 17-21 and 23 depend directly or indirectly from independent claim 16 and incorporate all of the limitations thereof. Accordingly, for at least the reasons established in section II above, Applicants respectfully submit that claims 17-21 and 23 are not anticipated by Furrer and respectfully request allowance of these claims.

Dependent Claim 17 includes limitations substantially similar to the limitations discussed in section I above. For example, as amended herein, Claim 17 recites:

> wherein the translator translates the data request into a generic format, and further comprising a data access layer . . . [that] determines, based on an enterprise data model, to direct the data request from one of the commercial-off-the-shelf software applications to the second data store, and translates the translated data request from the generic format into a storage format of the second data store.

Accordingly, the arguments of section I are hereby repeated for Dependent Claim 17.

In addition to the reasons established in section II above, for at least the reasons established in section I above, Applicants respectfully submit that all of the limitations of dependent Claim 17 are not anticipated by Furrer and respectfully request allowance of this claim.

Response to Rejections under Section 103

Claims Depending from Claim 16:

Claim 22 was rejected under 35 USC § 103(a) as being unpatentable over Furrer in view of Gajda, U.S. Patent No. 6,502,088 B1 ("Gajda").

Dependent claim 22 depends directly or indirectly from independent claim 16 and incorporates all of the limitations thereof. Accordingly, for at least the reasons established in section II above, Applicants respectfully submit that claim 22 is not taught or suggested by Furrer and respectfully request allowance of these claims. Gajda does not cure the deficiencies of Furrer.

CONCLUSION

Applicants respectfully submit that the pending application is in condition for allowance for the reasons stated above. If the Examiner has any questions or comments or otherwise feels it would be helpful in expediting the application, he is encouraged to telephone the undersigned at (972) 731-2295.

The Commissioner is hereby authorized to charge payment of any further fees associated with any of the foregoing papers submitted herewith, or to credit any overpayment thereof, to Deposit Account No. 21-0765, Sprint.

Respectfully submitted,

Date: MAY 28, 2009

John Leonard
Reg. No. 58,144
for
Michael W. Piper
Reg. No. 39,800

ATTORNEY FOR APPLICANTS

CONLEY ROSE, P.C.
5700 Granite Parkway, Suite 330
Plano, Texas 75024
(972) 731-2288
(972) 731-2289 (facsimile)